C O L O R C U E

Vol II.3 MAR'79 $1.00

READERS --

Most of you still have a few months to go on your 'no-cost six-months trial subscription' to ColorCue, but for others it's time to renew! If you've been enjoying the articles and information in this publication, then you'll want to fill out the form on the last page of this issue so that you won't miss a single ColorCue. You can determine when your subscription expires by looking at the number in the upper right of the mailing label. The number represents the date of the last issue of ColorCue that will be sent to you unless you renew your subscription. Those of you who see a 3-79 up there should renew now. We need to receive your form during the month of April in order to keep your subscription continuous. We'd like to be able to send each of you a reminder when your subscription runs out, but that's very costly, and you're probably tired of junk mail anyway! Just keep that expiration date in mind so you'll know when to to renew. If the date shown on your label seems to be incorrect, please write to us.

Since this is our first solicitation for subscriptions, we thought it only fitting that we should have the proverbial 'introductory offer'. We will charge exactly half of the listed $1.00 per issue price to anyone who renews a subscription during the next few months. After midsummer, this offer will expire, and subscription price will be back to the posted $1.00 per issue level. Our costs are considerably more than fifty cents per issue, and because we take on no advertising, ColorCue generates no revenue. Postage alone is at $.28 per copy, and paper and printing costs add more than $.22 to this. This introductory rate of $6.00 per year is applicable to all users in the United States, Canada, and Mexico.

****

This Issue's MENU

****

For those of you who live in Europe, Asia, Africa, etc., the ColorCue subscription rate is significantly higher -- you can renew at $1.50 per issue; $18.00 per year. We regret these high rates, but first class postage to other continents is about $1.25 per copy! We could send it third class and reduce the rate, but it has been our experience that third class mail can take **months** to reach its destination overseas. Overseas residents should check the appropriate space on the order blank on the last page.

READERS REQUEST
Program Appending

OK, David Suits of Rush, New York, HERE IT IS! That information we promised you about program appending. Since publishing David's request, we've heard from a lot of others who are on the edge of their seats waiting for this one.

It is easy to append two BASIC programs on the COMPUCOLOR II. The only requirements are that the line number ranges of the two programs are distinct, ie., 10 to 3000 and 10000-19000, and that both programs fit into memory at the same time. {If the line numbers are not distinct, a 'Program Merging' facility must be written in Assembly language which looks at the programs line by line. This is complicated to write, but some of you are undoubtedly capable of accomplishing it. The listing would be too long to include in ColorCue, but if there is enough demand, we may put such a 'MERGE' program on a utility Sof-Disk.}

Assume that you have two programs which fulfill the above mentioned requirements: LOW.BAS (with the low line numbers) and HIGH.BAS (with the high line numbers). The steps to append the programs are as follows:

1. Enter FCS by typing ESCAPE D.

2. Display the directory of the Sof-Disk containing LOW.BAS and look at the SADR (start address) entry in the directory. Subtract 2 from this number (use hexidecimal arithmetic!) and write the number down. For example, lets say it's B191.

3. Type LOAD LOW.BAS to load the first program into memory.

4. Next, type LOAD HIGH.BAS B191 The B191 is the hex number you calculated from LOW.BAS's SADR in step 2.

5. Re-enter BASIC by ESCAPE E. Now type LIST and both halves should become whole. You can now save the combined program by typing SAVE"NEW".

This process can be generalized to do multiple appends by simply noting the following facts:

1. All BASIC prorams load into memory starting at 829A hex.

2. The length of a BASIC program can be determined 2 ways from the directory entry of the program.

   a) Compute SADR - LADR to determine the length (in hex).
   b) Compute (SIZE-1) * 80 + LBC (in hex)

3. All BASIC programs have 2 zero bytes at the end to denote the physical end of the program.

4. Do not include the last 2 zero bytes when appending programs, except for the last program to be appended. The 2 zero bytes in the last program appended are used to denote the end of the newly constructed program.

KEEPING IT SIMPLE

Colors

Did you know that there's a way to get more than 8 colors on the COMPUCOLOR II? This can be done by using the null character, which is also called 'half-tone'. This charcter is the 'DELETE CHAR' key. If you press that key you will see that the character looks something like a checker-board, with alternating blank and colored squares. When you use this character, you can combine colors to get lots of different tints. In fact, you can get a total of 28 additional varieties! The total number of possibilities of 8 colors in any two combinations can be expressed as:

$$_8C_2 = \frac{8!}{2!(8-2)!}$$

We have two programs that demonstrate this ability, the first simply cycles through all the different possible combinations. We have commented it thoroughly so you can follow each program step.

The second program was written by a user in Omaha; Dennis Martin. He asked that we include his address and telephone number in his program comments, because he is interested in corresponding with other users. He says that there aren't a lot of COMPUCOLOR II's in the Omaha area, and he'd like a chance to talk over thoughts and ideas with other owners.

Since these colors are created with characters, they can only be used to fill in rectangular areas. But you may be able to achieve some interesting displays with this new range of hues.

```
100 PLOT 12,27,24                Erase page, page mode
200 B=16:F=16                    variable for background starts black
300 REM                          variable for foreground starts black
1000 FOR X=0 TO 61               this loop moves the line X of X-Y cursor
2000 FOR Y=0 TO 31               this loop draws the line Y of X-Y cursor
3000 PLOT 30,B                   background, B
4000 PLOT 29,F                   foreground, F
5000 PLOT 3,X,Y,96               position cursor, use character 96
6000 NEXT                        end of drawing one line
6100 REM
6250 REM                         draw three lines before changing color
6400 REM
6500 IF FLG<>3 THEN FLG=FLG+1:GOTO 10000
6550 REM
6600 FLG=0                       reset flag
7000 B=B+1                       change background
8000 IF B=24 THEN F=F+1:B=16     If all background colors done,
8100 REM                         increment foreground
9000 IF F=24 THEN END            If all foreground colors done,
9100 REM                         reset foreground
10000 NEXT
```

S P E C T R U M

```
10 REM  WRITTEN BY DENNIS L. MARTIN
20 REM 4817 SAHLER ST. APT. 9
30 REM OMAHA, NE.   68104
40 REM 402-453-6826
100 DIM E(50),F(50)
110 FOR G=1 TO 36
120 READ A,B
130 E(G)=A:F(G)=B
140 NEXT
150 RESTORE 500
299 W=0
300 P=0:Z=1:Y=0:PLOT 12
301 READ A
305 IF A=0 THEN 410
320 FOR X=P TO (P+10)
325 FOR L=Y TO (Y+1)
330 PLOT 3,X,L,30,A,32
340 NEXT:NEXT
350 READ B
360 FOR X=P TO (P+10)
365 FOR L=Y TO (Y+1)
370 PLOT 3,X,L,29,B,96
380 NEXT:NEXT
385 PLOT 25,25,25,30,16,29,23,14:PRINT "-";Z:PLOT 15
390 Y=Y+2
395 Z=Z+1
396 W=W+1
397 IF W=15 THEN W=0:P=P+20:Y=0
400 GOTO 301
410 PLOT 3,42,18:INPUT "COLOR NUMBER  :  ";N
415 IF N>35 OR N<=0 THEN 410
420 PLOT 3,42,20:PRINT "FOREGROUND = ";:GOSUB 640
430 PLOT 3,42,22:PRINT "BACKGROUND = ";:GOSUB 560
440 GOTO 410
500 DATA 21,16,20,17,21,20,23,20,21,21,20,16,20,20,22,20
510 DATA 22,16,20,18,22,22,23,22,18,16,18,18,19,18,22,18
520 DATA 22,19,23,18,23,19,19,19,18,17,19,16,19,17,23,17
530 DATA 21,19,23,21,21,17,17,17,17,16,23,16,22,17,21,18
540 DATA 22,21,20,19,23,23,0,0
555 END
560 IF E(N)=16 THEN PRINT "BLACK   ":RETURN
570 IF E(N)=17 THEN PRINT "RED     ":RETURN
580 IF E(N)=18 THEN PRINT "GREEN   ":RETURN
590 IF E(N)=19 THEN PRINT "YELLOW  ":RETURN
600 IF E(N)=20 THEN PRINT "BLUE    ":RETURN
610 IF E(N)=21 THEN PRINT "MAGENTA ":RETURN
620 IF E(N)=22 THEN PRINT "CYAN    ":RETURN
630 IF E(N)=23 THEN PRINT "WHITE   ":RETURN
640 IF F(N)=16 THEN PRINT "BLACK   ":RETURN
650 IF F(N)=17 THEN PRINT "RED     ":RETURN
660 IF F(N)=18 THEN PRINT "GREEN   ":RETURN
670 IF F(N)=19 THEN PRINT "YELLOW  ":RETURN
680 IF F(N)=20 THEN PRINT "BLUE    ":RETURN
690 IF F(N)=21 THEN PRINT "MAGENTA ":RETURN
700 IF F(N)=22 THEN PRINT "CYAN    ":RETURN
710 IF F(N)=23 THEN PRINT "WHITE   ":RETURN
```

NOTE: TO CHANGE SHADING EFFECT CHANGE LINE 385 TO READ :

```
385 PLOT 25,25,25,30,16,29,23:PRINT "-";Z
```

ADJUNCT AUTHOR

Linked Lists

A. E. Williams (the A. is for Alfred, we don't know what the E. is for) is from San Luis Obispo, California, and his was the first article we recieved in response to last month's request. It is surprising that he could have come up with something as interesting and well-written as this in such a short time. However, we've learned that he makes his living as a programmer, and its just possible that he's had a little experience with computers . . .

Alfred asked that instead of the 'Sof-Disk of his choice', we extend his ColorCue subscription in return for this article, and we have happily obliged. His article and the accompanying listing appear on the next pages.

****

COMMENTS AND CORRECTIONS

Ah yes, the good old corrections department. Just when we thought we were doing pretty well, the gremlins inserted a space in line 905 of the DUP program that appeared in January's issue. Line 905 was listed as

```
905 H$=" "
```

and it should have been

```
905 H$=""
```

It has also been suggested that it is not clear what the DUP program is asking for at lines 165 and 173. Change these to read:

```
165 PLOT 6,2:INPUT"INSERT SOURCE DISK AND HIT RETURN";Z:PLOT 27,4
173 PLOT 6,6:INPUT"INSERT DESTINATION DISK AND HIT RETURN";Z:PLOT 27,4
```

Now when you run the program, it will ask for exactly what it wants. Just hit return after you've put in the disk and closed the drive door.

Glenn Hayhurst of Denver, Colorado wrote to let us know about an error in December ColorCue. On page 5, the example that is supposed to draw a rectangle is missing one DEL. There should be four of them, one after the other. As it was printed, it draws a very charming right triangle.

The scrolling patch routine that appeared in October ColorCue needs a correction. Line 65190 is listed as `CLEAR 25:GOTO 1`
and it should be: `CLEAR 50:GOTO 1`
The program needs some amount of space over 32 in order to work properly.

We announced the formation of a local users group in the San Francisco area, and our coordinating dealer, Intersell, has asked us to let you know about a change in their mailing address. If you're interested in meeting and talking with other users in the Bay area write to the address below:

Intersell
540 Weddell Drive #9
Sunnyvale, CA 94086
ATTN: Mark Nehamkin

LINKED LISTS
(Part I: One-Way Linked Lists)

Linked lists provide a technique for making additions to a diskette file which is in ascending or descending sequence by a data item or combination of items without recopying the file. For example, suppose my family budget file contains a list of accounts numbered from 100 to 1000 in ascending sequence, and that the first digit of each account number determines what expense group an account belongs to (e.g., 100-199 are household expenses, 200-299 are for food, etc.)

Now suppose I hire that maid my wife has been hinting for for so long. I would like to insert a new record for account 121, "Domestic Help," after my current account 120 but before 200 (since it's a household-type expense.) With an ordinary sequential file, my only option is to recopy the entire file, inserting account 121 as I pass its position in the copy.

With a linked list, however, I can insert the account in logical sequence after account 120 without recopying the file. I do, however, pay a price, since linked lists require:

1) A little extra space in each record;
2) Enough space for required additions when the file is created; and
3) Some extra processing time to read the file sequentially.

Linked lists may be one-way or two-way linked. Two-way linked files permit relatively easy logical removal of records from file when they are deleted; with one-way linking, deleted records are ordinarily left on file, but flagged in some distinctive way to show that they are inactive. Two-way linking can be combined with a free list (a linked list containing all free (unused or deleted) records in the file space) to permit reclaiming the space left by deleted records. This article will consider a simple, one-way linking scheme. Part II, which will appear later, will expand the concept to describe two-way linked lists and the free list.

Basically, a one-way linked list requires addition of an extra variable (the link field) into each record. When the file is created, all link fields are set to zero except that in the last record; the latter is set to a number chosen to represent end of file (e.g., a negative number.) When a record is added, the link field of the preceding sequential record is examined; if it is zero, this record's number plus one is placed in the link field of the new (added) record. If not, the link field of the preceding logically sequential record is moved to the link field of the new record. In either case, the record number of the added record is placed in the link field of the preceding record, replacing its previous value.

A dummy record is placed at the beginning of the file, for two purposes:

1) To keep track of the next available empty space at the end of the file area, for additions; a field of this record should be set to the record number of the first record after the last record on file when it is initially built; as each addition in done, this counter is bumped by 1 to indicate the next available record. Good programming technique suggests that you should compare this number with the maximum file size in records (which is available from the FILE "A" command) in order to "die gracefully" if the file fills up.

2) To provide a link field to point to additions made before the first record on file.

Diagram 1 shows how a file originally created with records B, E, and F would be structured as a linked list before and after the addition of records A, C, and D.

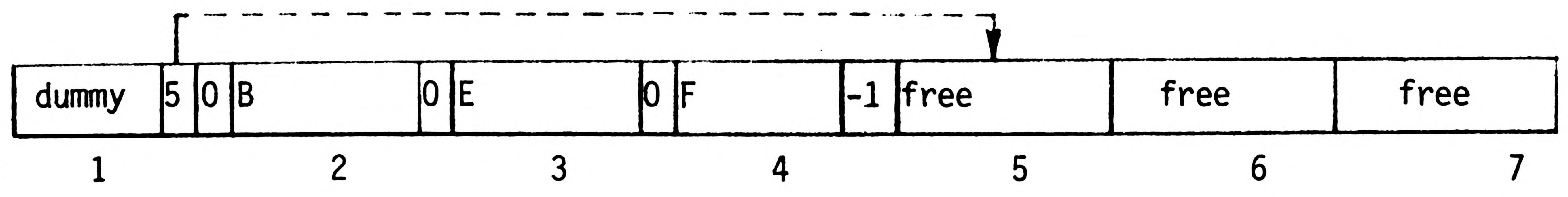

Diagram 1a - before addition of records

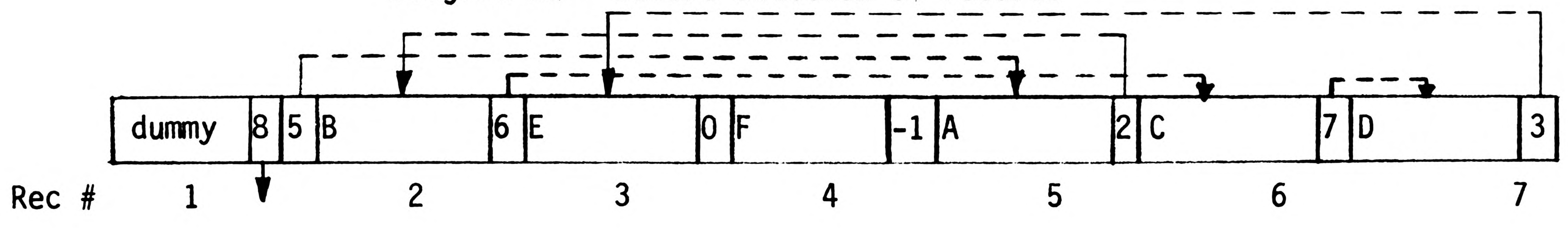

Diagram 1b - after addition of A, C, and D

In this diagram, the file is now out of space.

To process the file sequentially, we must ignore the dummy except to look at its link field. For all records, the next record to be returned is:

1) The next sequential record if the link field of the current record is zero;
2) Record number n if the link field has a value of n ≠ 0;
3) None (no more records) if the link field contains the file end value.

As an example, let's look at the file of Diagram 1. As you can see, it was created with space for three additional records at the end, all of which are used in diagram 1b. In processing sequentially, we first read record 1. Since it's the dummy, we'll ignore all of it but its link field. The latter contains a "5", so the first real record on file is record 5, i.e., the A record; after we've processed the latter, we then look at its link field in turn, and get record 2 as a result. Continuing, we reach record E, which has a link field of zero. This tells us to get the next sequential record, in this case 4. When we're done with 4 (F), we look at its link field, noting the negative number. Since this indicates end of file, we know we're done processing the file.

Now, what about deletions? When a record is deleted, it is flagged distinctively (a negative account number would do very nicely for my budget file.) If, during retrieval, I find a record thus marked, I ignore it except for its link field and get the next record.

Complicated? A little, but worth the effort if the file is large and therefore time-consuming to recopy. Linked lists are the basis for a number of more complicated file structures.

If the file is as small as the one in diagram 1, it could be recopied so quickly that the overhead and programming complexity of a linked list would be a waste. For a large file with many additions, however, a linked list can provide tremendous time savings (although you must also consider periodically <u>reorganizing</u> the file by recopying it to remove deleted records and pull active records back into physical sequence, setting the link fields to zero. This is desirable because buffering for physically sequential records is more efficient than for only logically sequential records, and some CPU time is required to process the links.)

I have included a BASIC listing for one-way linked list processing created as an example for this article. It can be used with few changes (by changing the composition of the record in the input, GET, PUT, and print routines,) but I would recommend studying the coding carefully to determine what it is doing and why. Understanding the underlying principles will help you in understanding more complex structures based on the simple linked list.

```
LIST


10 REM  LINKED LIST DEMO MAINLINE
20 CLEAR 128:FO= 0:PLOT 12
40 PRINT "WANT TO:":PRINT "1 - CREATE FILE":PRINT "2 - PRINT FIL
E":PRINT "3 - ADD A RECORD"
50 PRINT "4 - DELETE A RECORD":PRINT "5 - END THE RUN?":INPUT R
120 IF R< 1OR R> 5THEN 40
160 ON RGOTO 200,600,720,760,800
199 REM  FILE CREATION ROUTINE
200 INPUT "HOW MANY RECORDS, INCLUDING ADDS?";FS:IF FS< 1THEN 20
0
240 FILE "N","EXAMPL",FS+ 1,28,4:GOSUB 2600:OK= - 1
280 PRINT "ENTER RECORDS IN ASCENDING SEQUENCE BY KEY":I1= 2
320 GOSUB 2700:IF KE< = OKTHEN PRINT "SEQUENCE ERROR ON KEY":GOT
O 320
360 OK= KE
400 INPUT "LAST RECORD? (Y/N)";R$:IF R$< > "Y"AND R$< > "N"THEN
400
440 IF R$= "Y"THEN 520
480 LN= 0:GOSUB 1900:I1= I1+ 1:GOTO 320
520 LN= - 1:GOSUB 1900:REM  LAST RECORD WRITTEN - NOW SET UP DUM
MY
540 FR= I1+ 1:LN= 0:DA$= "***DUMMY***":PUT 1,1,1;FR,DA$[20],LN:G
OTO 40
599 REM  FILE PRINT ROUTINE
600 EF= 1:GOSUB 2600
640 GOSUB 1100:IF EF< > 1THEN PRINT KE,DA$:GOTO 640
680 GOTO 40
719 REM  ADD A RECORD ROUTINE
720 GOSUB 2600:GOSUB 2700:GOSUB 2290:GOTO 40
739 REM  DELETE RECORD ROUTINE
760 GOSUB 2600:GOSUB 2900:GOSUB 2050:GOTO 40
799 REM  END OF PROCESSING - CLOSE FILE
800 FILE "C",1:END
1000 REM  GET RECORDS SEQUENTIALLY. SET EF=1 WHEN RETURNING
1010 REM  LAST RECORD ON FILE.(EF SHOULD BE 1 FIRST TIME
1020 REM  ROUTINE IS ENTERED. VARIABLE USE: I1 -> RECORD
1030 REM  NUMBER FOR FCS; KE -> KEY VARIABLE (FILE IS IN
1040 REM  ASCENDING SEQUENCE BY KE); LN -> LINK FIELD FROM
1050 REM  CURRENT RECORD; LR -> REC # OF OLD LOGICAL RECORD.
1100 IF EF< > 1THEN 1250:REM  NOT 1ST ENTRY TO ROUTINE OR EOF
1200 EF= 0:I1= 1:GOSUB 1800
1250 LR= I1:IF LN> 0THEN 1400
1300 IF LN= 0THEN I1= I1+ 1:GOTO 1450
1350 EF= 1:RETURN
1400 I1= LN
1450 GOSUB 1800:IF KE< 0THEN 1250
1470 RETURN
1500 REM  SEARCH FOR A SPECIFIC RECORD. KEYS MUST BE UNIQUE AND
1510 REM  ASCENDING. SET NF=1 IF RECORD NOT FOUND.
1520 REM  VARIABLE USAGE: KE -> KEY; LR -> RECORD NUMBER OF
1530 REM  OLD RECORD. NF -> 0 IF RECORD FOUND, 1 IF RECORD NOT
1540 REM  FOUND. SK -> SAVE KEY TO BE LOCATED.
1550 NF= 0:SK= KE:EF= 1
1580 GOSUB 1100
1610 IF EF= 1OR SK< KETHEN NF= 1:RETURN
1640 IF KE= SKTHEN RETURN
1670 GOTO 1580
1790 REM  READ A RECORD NUMBER I1 FROM THE RANDOM FILE
1800 GET 1,I1,1;KE,DA$[20],LN
1850 RETURN
1890 REM  WRITE A RECORD I1 TO THE RANDOM FILE
1900 PUT 1,I1,1;KE,DA$[20],LN;RETURN
2000 REM  DELETE A RECORD OF KEY KE
2010 REM  DELETES ARE THOSE HAVING A KEY ( 0. THEY ARE IGNORED
2020 REM  BY THE GET SEQUENTIAL ROUTINE (LN 1000 ET SEQ).
2050 GOSUB 1550;IF NF= 1THEN RETURN
2100 KE= - 1;DA$= "***DELETED RECORD***"
2150 GOSUB 1900;RETURN
2200 REM  ADD A RECORD AFTER FILE IS BUILT. NEW KEY IS ASSUMED
2210 REM  TO BE IN KE, NEW DATA IN DA$. THE SEARCH ROUTINE IS
2220 REM  USED TO VERIFY THAT THE REC IS NOT ALREADY ON FILE, AN
D
2230 REM  TO FIND THE PRECEDING LOGICAL RECORD.
2240 REM  THE ROUTINE THEN DOES THE CORRECT LINK FIELD SWAPPING
2250 REM  FOR AN ADD. VARIABLE USAGE SD$ -) SAVE DA$ DATA;
2260 REM  SK -) SAVED KEY (DONE IN SEARCH RTN);FR -) FIRST AVAIL
ABLE REC (FROM DUMMY)
2270 REM  SR -) SAVE FOR FR; LN -) LINK FIELD FROM CURRENT REC
2280 REM  SL -) SAVE FOR LN; FS -) FILE SIZE FROM FILE "A"
2290 SD$= DA$
2300 GOSUB 1550;IF NF= 0THEN PRINT "RECORD ";SK;" ALREADY ON FIL
E";RETURN
2350 GET 1,1,1;FR;SR= FR;FR= FR+ 1;PUT 1,1,1;FR
2360 REM  THE PRECEDING LINE HAS RETRIEVED & UPDATED THE FIRST
2370 REM  AVAILABLE RECORD NUMBER ON THE DUMMY.
2400 I1= LR;GOSUB 1800;SL= LN;LN= SR;GOSUB 1900
2410 REM  THE PRECEDING PLUGS LINK FIELD FOR THE NEW RECORD
2420 REM  INTO THE PRECEDING RECORD & REPLACES IT.
2450 KE= SK;DA$= SD$;LN= I1+ 1;IF SL( ) 0THEN LN= SL
2500 I1= SR;GOSUB 1900;IF FR) FSTHEN PRINT "NO MORE ADD SPACE -
REORGANIZE FILE"
2510 REM  THE PRECEDING WRITES THE NEW RECORD & CHECKS FILE SPAC
E
2550 RETURN
2599 REM  FILE OPEN ROUTINE
2600 IF FO= 1THEN RETURN
2650 FO= 1;FILE "R",1,"EXAMPL",3
2680 FILE "A",1,DU,FS,DU,DU;RETURN
2690 REM  ACCEPT A RECORD FROM TERMINAL
2700 GOSUB 2900;INPUT "ENTER TEXT DATA";DA$;RETURN
2890 REM  ACCEPT KEY FROM TERMINAL
2900 INPUT "ENTER KEY ()0)";KE;IF KE( = 0THEN 2900
2950 RETURN

READY
```

COMPOSITE COLOR CODES

On the following page is a list of the color codes used in plot statements. Our programmers here at Compucolor Corporation use this list as a quick reference, and you may find it useful, too. It saves having to make the calculations or memorizing the codes for these color combinations. We've put it on a separate page so that you can tear it out and put it in your manual.

MANUAL INDEX

Robert Olesen, of Belmont, California, is the owner of a shiny new COMPUCOLOR II, Model 4. He reports that while using the Programming Manual, he had some trouble locating all of the information. Well, we think our table of contents is pretty complete, but we have to admit that this index Robert wrote is even better! We appreciate the time and effort that went into making this index, and we thought that it might be useful for all Programming and Reference Manual owners. We hope you're all Manual owners by now, since the Manual contains information that will make using your COMPUCOLOR II even easier.

****

Did you know that there is a Maintenance Manual available from Compucolor Corporation? It contains schematics and circuit descriptions that will interest those of you with any knowledge of electronics. You can buy the Manual from your dealer, or you can order directly from Compucolor Corporation. The price is $50.00

Compucolor Corporation will be represented at a number of Electronics/ Computer shows in the next few months, and we hope that some of you will stop by to say hello. At the West Coast Computer Faire, we will have a 'Challenge the COMPUCOLOR II' Othello tournament, which is sponsored by our company and CBS toys. More specifics about this next month. In June, we will be in New York at the NCC.

We are pleased with the response we got from the call for articles that we had last month. We will try to feature at least one user-written article in each of the forthcoming issues of ColorCue. This month's article is a long one, and it has lots of good information presented in a way that makes it easy to understand. Next month, we will have an article about how you can make a dust cover for your machine, along with a long-awaited description of the Personal Data Base -- including what it is, what it does, and why you can't live without it!

****

COMPOSITE COLOR CODES

GROUNDS				GROUNDS			
FORE	**BACK**	**PLOT**	**BLINK**	**FORE**	**BACK**	**PLOT**	**BLINK**
BLACK	BLACK	0	64	BLACK	BLUE	32	96
RED	BLACK	1	65	RED	BLUE	33	97
GREEN	BLACK	2	66	GREEN	BLUE	34	98
YELLOW	BLACK	3	67	YELLOW	BLUE	35	99
BLUE	BLACK	4	68	BLUE	BLUE	36	100
MAGENTA	BLACK	5	69	MAGENTA	BLUE	37	101
CYAN	BLACK	6	70	CYAN	BLUE	38	102
WHITE	BLACK	7	71	WHITE	BLUE	39	103
BLACK	RED	8	72	BLACK	MAGENTA	40	104
RED	RED	9	73	RED	MAGENTA	41	105
GREEN	RED	10	74	GREEN	MAGENTA	42	106
YELLOW	RED	11	75	YELLOW	MAGENTA	43	107
BLUE	RED	12	76	BLUE	MAGENTA	44	108
MAGENTA	RED	13	77	MAGENTA	MAGENTA	45	109
CYAN	RED	14	78	CYAN	MAGENTA	46	110
WHITE	RED	15	79	WHITE	MAGENTA	47	111
BLACK	GREEN	16	80	BLACK	CYAN	48	112
RED	GREEN	17	81	RED	CYAN	49	113
GREEN	GREEN	18	82	GREEN	CYAN	50	114
YELLOW	GREEN	19	83	YELLOW	CYAN	51	115
BLUE	GREEN	20	84	BLUE	CYAN	52	116
MAGENTA	GREEN	21	85	MAGENTA	CYAN	53	117
CYAN	GREEN	22	86	CYAN	CYAN	54	118
WHITE	GREEN	23	87	WHITE	CYAN	55	119
BLACK	YELLOW	24	88	BLACK	WHITE	56	120
RED	YELLOW	25	89	RED	WHITE	57	121
GREEN	YELLOW	26	90	GREEN	WHITE	58	122
YELLOW	YELLOW	27	91	YELLOW	WHITE	59	123
BLUE	YELLOW	28	92	BLUE	WHITE	60	124
MAGENTA	YELLOW	29	93	MAGENTA	WHITE	61	125
CYAN	YELLOW	30	94	CYAN	WHITE	62	126
WHITE	YELLOW	31	95	WHITE	WHITE	63	127

ColorCue Subscription Renewal

United States; Canada; Mexico;

___ YES!! Please renew my subscription to ColorCue for 1 (one) year at the introductory rate of $.50 per issue. Enclosed is my check for $6.00 payable to 'Compucolor Corporation'

Europe; Asia; Africa; Australia; South America; Antarctica;

___ YES!! Please renew my subscription to ColorCue for 1 (one) year. I have enclosed a check for subscription at the overseas rate of $18.00. (U.S. dollars)

NAME (as printed on address label) ______________________________

Use the following lines for CORRECTIONS ONLY:

NAME ____

ADDRESS ____

CITY ____________________ STATE ______________ ZIP _________

COMMENTS OR SUGGESTIONS: _________________________________

____

____

____

____

____

Mail to: Compucolor Corporation,
P.O. Box 569,
Norcross, GA 30071
U.S.A.

ATTN: ColorCue Subscription Department